UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

SYLLABUS DEL CORSO

Machine Learning for Finance

2627-1-F1602M003

Learning objectives

Knowledge and understanding. The course aims to provide students with solid skills in Python programming and with the fundamental concepts of data processing and machine learning: the representation of data through vectors, matrices and tables, models that learn from data, and artificial neural networks.

Applying knowledge and understanding. At the end of the course, students will be able to write Python programs that read and process financial and economic data, simulate random phenomena, analyse tabular data with the NumPy and pandas libraries, represent data graphically, and build and train simple neural networks with PyTorch.

Making judgements. Students will learn to choose independently, among the tools and techniques studied, those best suited to solving practical problems of moderate complexity related to the manipulation and analysis of financial data, and to assess critically the correctness of the results obtained.

Communication skills. Students will be able to describe clearly and rigorously, both orally and in writing, algorithms, programs and machine learning models, using appropriate terminology and writing readable, well-documented code.

Learning skills. Students will acquire the skills needed for independent and lifelong learning in programming and data analysis; in particular, they will be able to consult the official documentation of languages and libraries in order to explore new tools and apply them to problems different from those addressed in the course.

Contents

The course provides solid skills in Python programming, from the fundamentals of the language to the libraries for numerical computing (NumPy), tabular data analysis (pandas) and plotting (matplotlib), and introduces artificial neural networks, implemented with PyTorch, as a tool supporting the analysis of financial data.

Detailed program

1. Algorithms and programs: definition and properties of an algorithm; problems, instances, data and results; from algorithms to programs; programming paradigms.

2. The PyCharm development environment: the Python interpreter, projects and files, running a program, the Python Console, the debugger. The lexicon of the language: keywords and identifiers.

3. Data types and collections: integers and floating-point numbers, booleans, strings, None, type conversions; arithmetic, relational and logical expressions; lists, tuples, sets and dictionaries.

4. Compound statements: selection (if, elif, else), for and while loops, the break and continue statements.

5. Pseudo-random numbers and simulation: the random module, discrete distributions and the normal distribution, simulation of random phenomena, Markov processes.

6. Text files: opening and reading with open and with; files with several fields per line; header lines; building tables from the data read; parameters read from a file or from the keyboard.

7. Functions: definition and call, type annotations, parameters and arguments, return values, local and global variables, parameter passing, recursion, modules and import.

8. The NumPy library: vectors and matrices, indexing and submatrices, element-wise and vector operations, summary functions, boolean masks, elements of linear algebra, reading matrices from files, random numbers; vectors and matrices in machine learning: linear regression and gradient descent.

9. The pandas library: Series and DataFrame; reading data with read_csv and read_excel; selecting rows and columns with loc, iloc and masks; common labels (isin, intersection); arithmetic and relational operations and label alignment; summary statistics; aggregation with groupby and agg; transposition; sorting, adding, renaming and dropping rows and columns; combining tables (concat, merge, join); missing values; categorical variables; cross tabulations; time series; normalisation and standardisation; covariance, correlation and regression line.

10. Plotting with matplotlib: line plots, scatter plots, bar charts and histograms; multiple series, legends and styles; several plots in one figure; pandas plotting.

11. Classes and objects: defining a class, methods, class and instance attributes, special methods, inheritance and method overriding, abstract classes, the **call** method.

12. Artificial neural networks: the artificial neuron and the perceptron; activation functions; the multilayer perceptron; training a network with gradient descent and error backpropagation. Implementation with PyTorch: tensors (torch.tensor), automatic differentiation (backward), optimisers (torch.optim, step, zero_grad), layers and the network class (torch.nn.Linear, torch.nn.ReLU), loss functions, Dataset and DataLoader, the training loop, dropout and batch normalisation (torch.nn.Dropout, torch.nn.BatchNorm1d), data preparation; introduction to deep networks.

Prerequisites

The mathematical and logical knowledge acquired in secondary school; basic notions of linear algebra (vectors,

matrices, dot product) and of statistics (mean, variance, probability distributions). Basic Python programming skills are also required: data types, selection and loop statements, functions.

Teaching methods

Lectures with guided exercises. Lessons take place in the teaching laboratories, so that students can immediately apply the concepts presented on the computer.

Assessment methods

The exam consists of a written test, held in the University's teaching laboratories, in which students develop Python programs to solve data processing and analysis problems. The test covers all the topics of the course: basic Python programming, data processing with the NumPy and pandas libraries, and artificial neural networks. To pass the exam, students must demonstrate sufficient knowledge of each of these topics; a serious gap in one of them cannot be compensated by the results obtained in the others. The assessment considers the correctness of the results, the ability to choose the tools appropriate to the problem, and the clarity of the code. During the test, only the official documentation of Python and of the libraries used may be consulted.

Textbooks and Reading Materials

Paul J. Deitel, Harvey M. Deitel, **Intro to Python for Computer Science and Data Science**, Pearson (Italian edition: Introduzione a Python. Per l'informatica e la data science, Pearson, 2021, ISBN 978-8891915924).
Lecture notes, examples and exercises provided by the lecturer on the course e-learning platform.
Official documentation: docs.python.org, numpy.org/doc, pandas.pydata.org/docs, matplotlib.org, pytorch.org/docs.

Semester

First semester

Teaching language

Italian

Sustainable Development Goals

QUALITY EDUCATION